Schema Mapper: Implementation Log

overview

A system that converts text to JSON following any schema. Uses Azure OpenAI GPT-4 + Streamlit web interface.

Development Process

Phase 1: Basic Setup

Files: main.py, logger.py, .env

- Researched best LLMs for JSON output and handling large schemas (e.g., GPT-4, Claude 3, Gemini 1.5, JSONFormer(huggingface))
- Compared model capabilities: context length, JSON mode reliability, and schema adherence
- Finalized basic project structure with modular design for model calls, parsing, and schema validation
- Set up connection using Azure AI Foundry to access OpenAI's GPT-4
- Created logging system with timestamped files

Phase 2: LLM Configuration

Experiments:

- Temperature: Tried 0.1 (too rigid) → 0.8 (too creative) → 0.4 (perfect balance)

- JSON Mode: Reduced non-JSON responses by 95%

- Top-p: Set to 0.9 for quality balance

Phase 3: Prompt Engineering

Evolution:

1. Basic: "Convert text to JSON" → Failed often

2. Detailed: Added numbered steps → Better compliance

3. Final: Explicit data type rules → Best Results

Current Prompt Strategy:

- Clear numbered instructions.

- Explicit boolean handling.

- Schema adherence enforcement.

Phase 4: Testing

All three test cases were successfully passed:

1. Academic Paper (BibTeX → Citation JSON)
2. GitHub Actions (Description → Config JSON)
3. Resume (Text → Structured JSON)

Phase 5: Web Interface

File: app.py

- Built a minimal Streamlit app
- Enabled file upload and side-by-side input/output view
- Added basic download functionality.